Booster Extraction Losses

Local application

Tue, Dec 18, 2007

When tuning Booster to minimize beam losses during extraction, watching the integrated loss monitor waveforms can be tedious. When losses occur early in the cycle, the losses at the end of the cycle show the sum of both. A local application called `BLMD` is designed to capture the losses separately that occur around extraction. This note describes the approach used.

There are five loss monitors that are positioned to be sensitive to beam losses at extraction time. These integrated loss signals are digitized by the Swift digitizer at 12.5 KHz, or 80 μs per point. A local application called `BLMS` interprets these digitized waveforms and builds various sums of the losses of interest to users, just as it does for all 70 Booster BLMs. One type of sum is a "forever" sum of beam losses occurring in each BLM and each Booster reset clock event, for each millisecond since Booster reset. There are 39 millisecond sums, plus one total loss sum, that is maintained in nonvolatile memory as double precision values. This data can be used by examining differences between such millisecond sums between any two points in time. All of these sum data are captured every 2 minutes by the Acnet datalogger.

The new LA called `BLMD` is designed to build on these millisecond summed loss values to produce the losses near extraction. Suppose we assume that Booster extraction losses can be measured by the millisecond losses that occur 36–38 ms after the Booster reset event. We need to sum three of the ms losses. To do this, we need to know the losses that occur since the forever sums were accumulated on the previous 15 Hz cycle. To compute those losses, we need to look at the difference between the ms sums of the previous cycle and those of the current cycle. Summing three of these differences, then, gives us the loss around extraction time.

More detail

During LA initialization, `BLMD` forms a data request for the millisecond loss sums that are needed to cover the BLMs of interest and the reset events of interest, since a separate result will be shown for each reset event taken from the series `0x11`, `12`, `13`, `14`, `15`, `16`, `17`, `19`, `1C`, `1D`, and `10`. (All but the first two are beam events.) These sums are alternately received into two buffers, so that `BLMD` always has access to the current cycle sum data as well as the previous cycle sum data. Besides the ms sums, there is also an array of event sums available, and these are included in the data request. Each cycle, then, this event sum array is checked for changes in the event counts. There will normally be two changes detected for each beam event. One will be the specific beam cycle, such as `0x1D`, and the other will be the any-beam-event `0x10`. For each such changed event count, the corresponding ms losses for that event are summed as stated above, and the result is deposited into the local data pool. With 9 beam events, and two BLMs, say, we need 18 new devices to hold these loss sums.

Parameter layout

Consider the following parameter list for `BLMD`:

Field	*Size*	*Meaning*
`ENABLE B`	2	Usual local application enable Bit#
`TARGNODE`	2	Target BLM node for data requests
`INIT BLM`	2	First BLM Chan#
`NCHANS`	2	Number of BLMs to access
`MS RANGE`	2	Initial (hi), final (lo) ms (range 1–40)
`EV RANGE`	2	Initial (hi), final (lo) event index (range 0–11)
`SUM C`	2	Local node Chan# for summed results as floats

Of the five BLMs of interest for Booster extraction, two (`IRML03` and `IRMS03`) are connected to `node06C3`, and three (`IRM023`, `IRM025`, and `IRM026`) are connected to `node06C6`. This means that `BLMD` will run in two nodes. If the channels in one node are not consecutive, there may be more than one instance of the LA in that node.

The result channel order is all chosen reset events for the first channel, all for the second, etc.

With the `TARGNODE` parameter, one can run `BLMD` for testing purposes in a separate node from the one that is connected to the BLMs. The data request will cause the BLM millisecond sums to be delivered over the network at 15 Hz. Because an LA is making the request over the network, the result data will be one cycle behind. But when `TARGNODE` is the local node, no network traffic is needed, and the data will be current and correlated. (There is nothing special in the code of `BLMD` to achieve this; it just falls out of the normal front end operation.)

To request the 9 beam events, the `EV RANGE` parameter is `0x020A`, implying the range of indices of the sequence of clock events listed above. To specify the ms sums in the range 36–38 ms, use `0x2426` for the `MS RANGE` parameter. For two channels, for which the sum data is obtained via the network from the target node, the execution time needed is 300 μs.